Performance Evaluation of the SX6 Vector Architecture for Scientific Computations

Leonid Oliker

Future Technologies Group

Computational Research Division

LBNL

www.nersc.gov/~oliker

Andrew Canning, Jonathan Carter, John Shalf, David Skinner: LBNL

Stephane Ethier: PPPL

Rupak Biswas, Jahed Djomehri, and Rob Van der Wijngaart: NASA Ames

Motivation

- ✍ Superscalar cache-based arch dominate US HPC
- ✍ Leading arch are commodity-based SMPs due to cost effectiveness and generality (and increasing peak perf)
- ✍ Growing gap peak & sustained perf well known in sci comp
- ✍ Modern parallel vectors offer to bridge gap for many apps
- ✍ Earth Simulator has shown impressive sustained perf on real scientific apps and higher precision simulations
- ✍ Compare single node vector NEC SX6 vs cache IBM Power3/4 for several key scientific computing areas
- ✍ Examine wide spectrum of algorithms, program paradigm, and parallelization strategies

Architecture and Metrics

Node Type	Name	CPU/ Node	Clock MHz	Peak GFlop	Mem BW GB/s	Peak B/F	MPI Lat usec
Power3	Seaborg	16	375	1.5	0.7	0.4	8.6
Power4	Cheetah	32	1300	5.2	2.3	0.4	3.0
SX6	Rime	8	500	8.0	32	4.0	2.1

Microbenchmark performance

- Memory subsystem, strided, scatter/gather w/ STREAM/XTREAM
- MPI: point-point comm, network contention, barrrier synch w/ PMB
- OpenMP: reduction and thread creation w/ EEPC

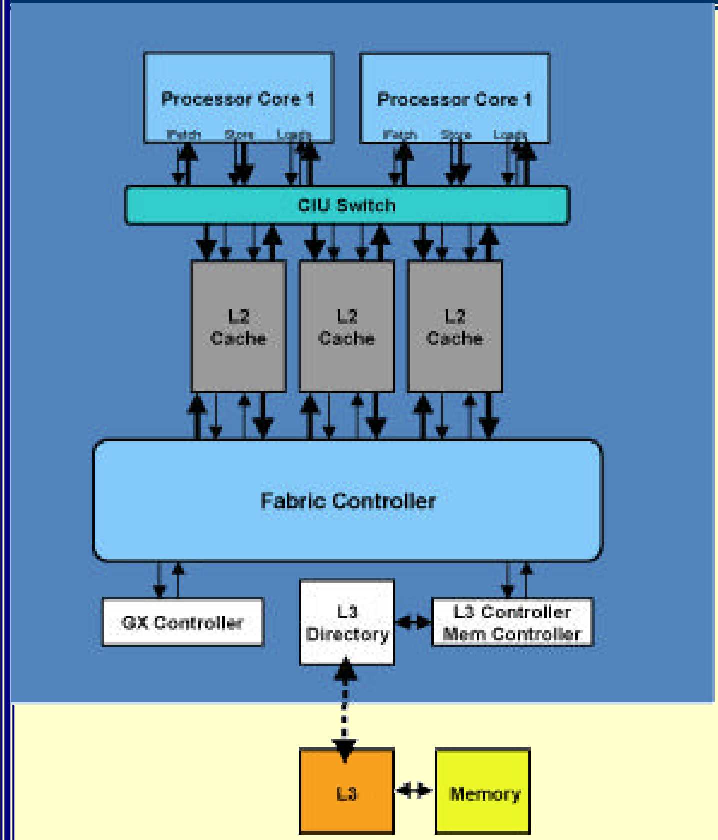
Application Performance

- CACTUS:Astrophysics - Solves Einstein's equations
- TLBE: Fusion - Simulations high temp plasma
- PARATEC: Material Science – DFT electronic structures
- Overflow-D: CFD – Solves Navier-Stokes equation around complex geometries
- GTC: Fusion – Particle in cell to solve gyrokinetic Vlasov-Poisson equation
- Mindy: Molec Dynamics – Electrostatic interaction using Particle Mesh Ewald

Power3 Overview

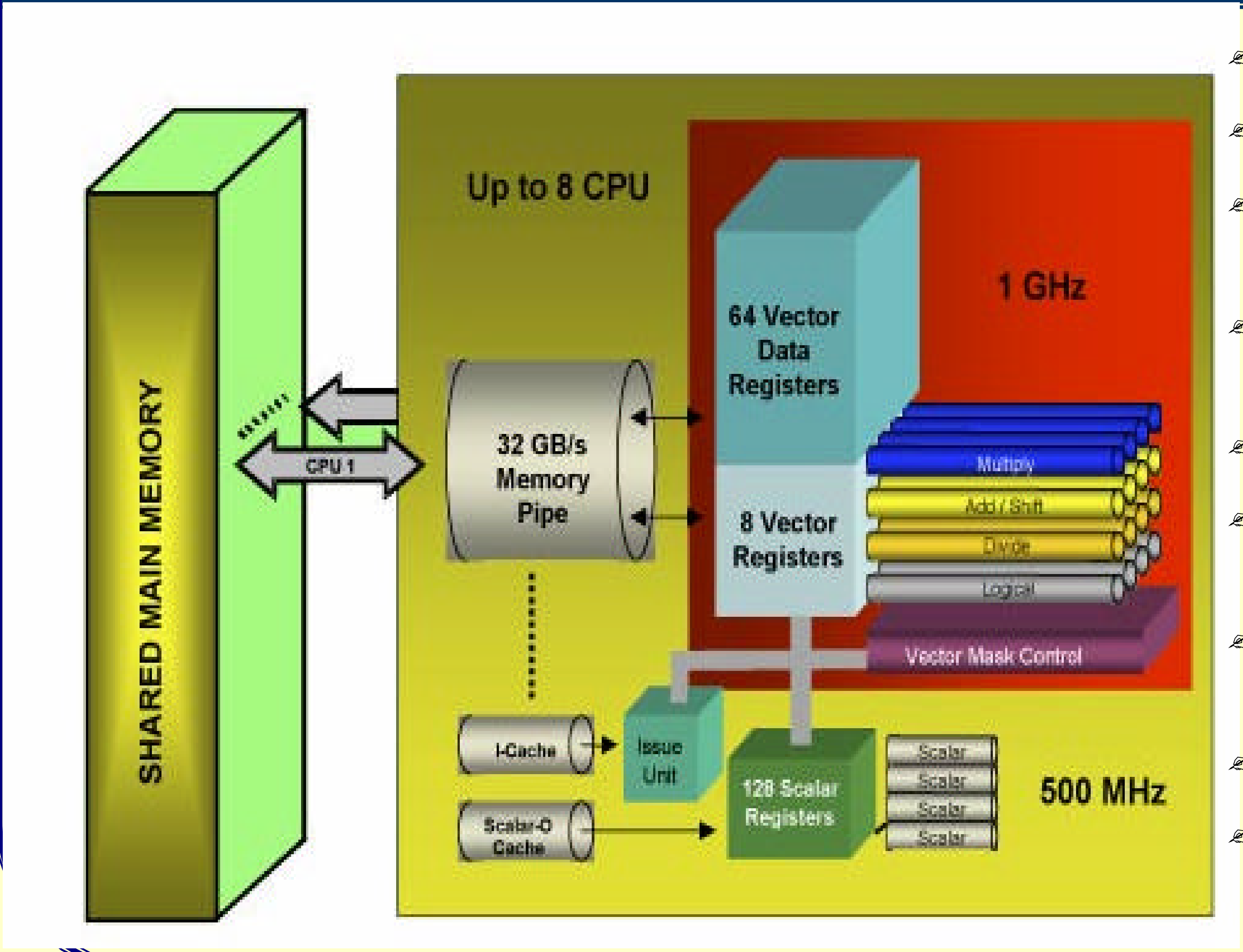
- 375 MHz procs w/ 2 FPU can issue MADD: peak 1.5 Gflops
- Short 3 cycle pipeline (low penalty branch misprediction)
- RISC, Peak 8 inst per cycle, sustained 4 inst per cycle
- Superscalar out-of-order w/prefetching
- CPU has 32KB Instr Cache and 128KB L1 Cache
- Off-chip 8MB 4-way set associative L2 Cache
- SMP node 16 processors connected to mem via crossbar
- Multi-node networked IBM Colony switch (omega topology)

Power4 Overview



- Power4 chip contains 2 1.3 GHz cores
- Core has 2 FPU w/ MADD, peak 5.2 Gflop/s
- 2 load/store units per core
- 8-way suprsclr o-o-o, prefetch, brnch predict
- RISC, 6 cycle pipeline
- Private L1 64K Inst C and 32K Data C
- Shard 1.5 MB unified L2
- L2s on MCM connected point-point
- 32 MB L3 off-chip, can be combined w/ other L3s on MCM for 128MB L3
- 32 SMP, 16 P4 chips, organized 4MCM
- Current Colony switch, future is Federation

SX6 Overview



- 8 Gflops per CPU
- 8 CPU per SMP
- 8 way replicated vector pipe
- 72 vec registers, 256 64-bit words
- MADD unit
- 32 GB/s pipe to DDR SDRAM
- 4-way superscalar o-o-o @ 1 Gflop
- 64KB I$ & D$
- ES: 640 SX6 nodes

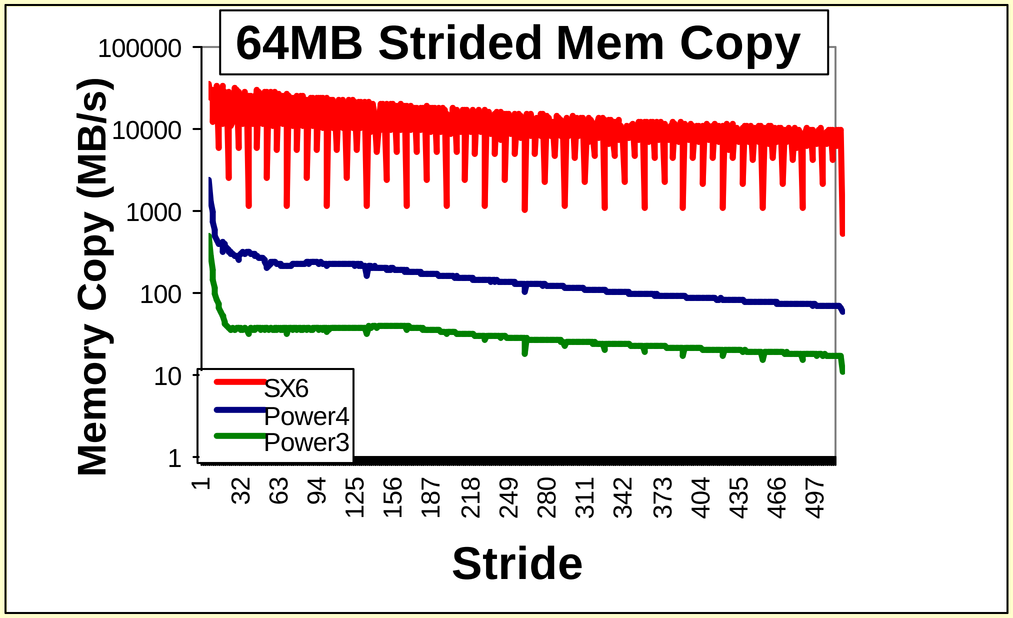
Memory Performance STREAM Triad

P	Power 3		Power4		SX6	
	GB/s	%Deg	GB/s	%Deg	GB/s	%Deg
1	0.66	0	2.29	0	31.9	0
2	0.66	0	2.26	1.2	31.8	0.2
4	0.64	2.6	2.15	6.2	31.8	0.1
8	0.57	14.1	1.95	15.1	31.5	1.4
16	0.38	42.4	1.55	32.3		
32			1.04	54.6		

$a(i) = b(i)+s*c(i)$

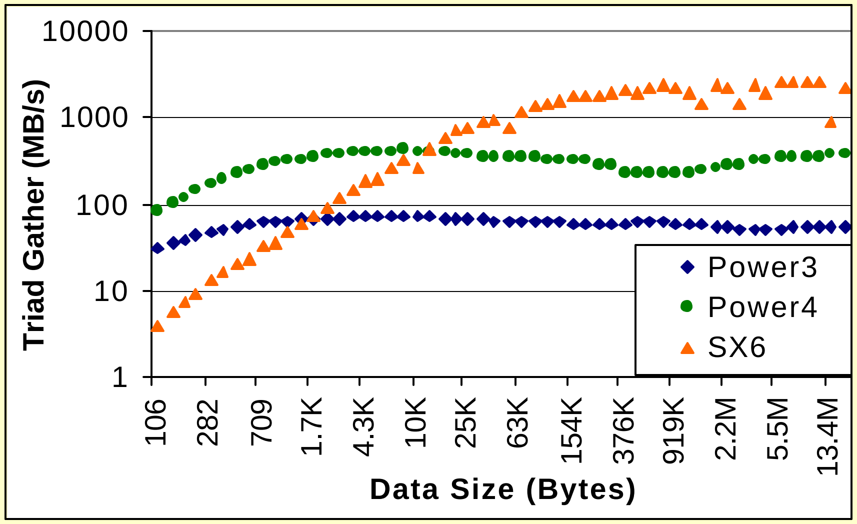
- Unit stride STREAM microbenchmark captures effective peak bandwidth
- SX6 achieves 14x and 48x single proc performance over Power3/4
- SX6 shows negligible bandwidth degradation,
  Power3/4 degrade around 50% for fully packed nodes

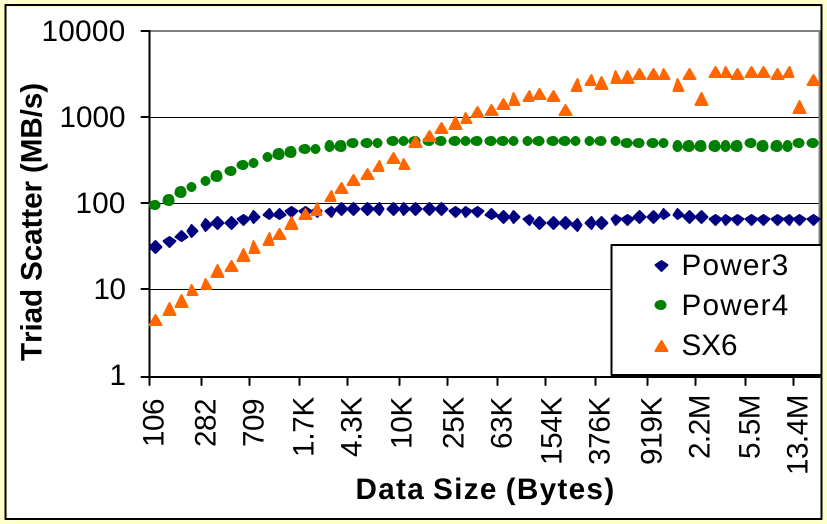
Memory Performance
Strided Memory Copy



- SX6 achieves 3 and 2 orders of magnitude improvement over Power3/4
- SX6 shows less average variation
- DRAM bank conflicts affect SX6 : prime #s best, powers 2 worst
- Power3/4 drop in performance for small strides due to cache reuse

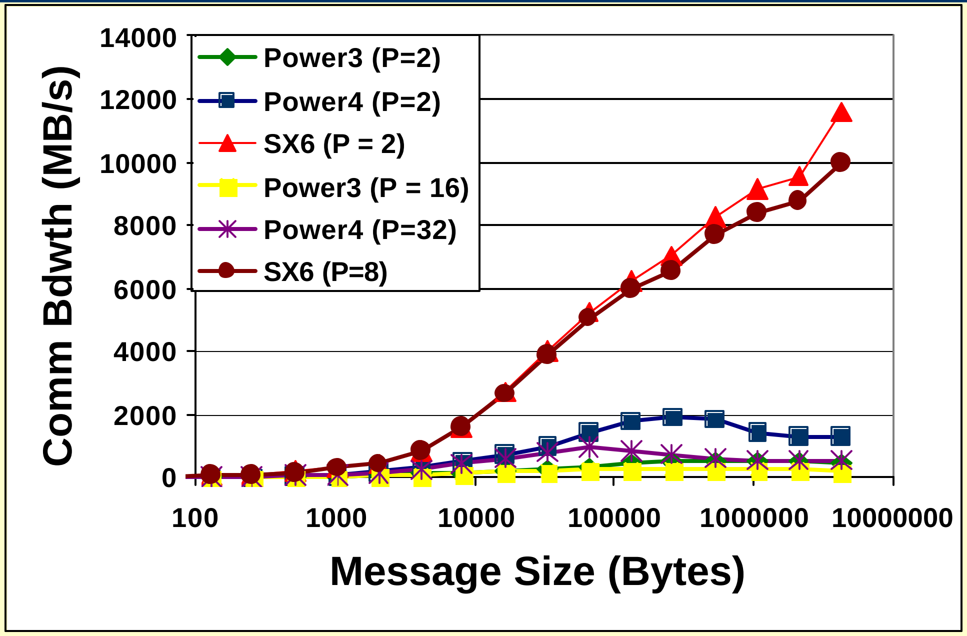
Memory Performance Scatter/Gather





- Small (in cache) data sizes Power3/4 outperform SX6
- Larger data sizes SX6 significantly outperforms Power3/4
- SX6 large data sizes allows effective pipelining & scatter/gather hdwr

MPI Performance Send/Receive



P	128KB			2MB		
	Pwr3	Pwr4	SX6	Pwr3	Pwr4	SX6
2	0.41	1.76	6.21	0.49	1.13	9.58
4	0.38	1.68	6.23	0.50	1.24	9.52
8	0.34	1.63	5.98	0.38	1.12	8.75
16	0.26	1.47		0.25	0.89	
32		0.87			0.57	

MPI Send/Receive (GB/s)

- For largest messages SX6 higher bdwth 27x Power3 and 8x Power4
- SX6 significantly less degradation with fully saturated SMP:
- Example at 2^{19} bytes w/ fully saturated SMP performance degradation:
  - Power3 46%, Power4 68%, SX6 7%

Synchronization and OpenMP Performance

P	MPI (*usec*)			OpenMP (*usec*)					
	Synchronization			Thread Spawning			Scalar Reduction		
	Pwr3	Pwr4	SX6	Pwr3	Pwr4	SX6	Pwr3	Pwr4	SX6
2	17.1	6.7	5.0	35.5	34.5	24.0	37.8	16.3	24.0
4	31.7	12.1	7.1	37.1	35.6	24.3	40.6	17.3	24.3
8	54.4	19.8	22.0	42.9	37.5	25.2	51.4	19.9	25.3
16	79.1	28.9		132.5	54.9		64.2	38.1	
32		42.4			175.5			158.3	

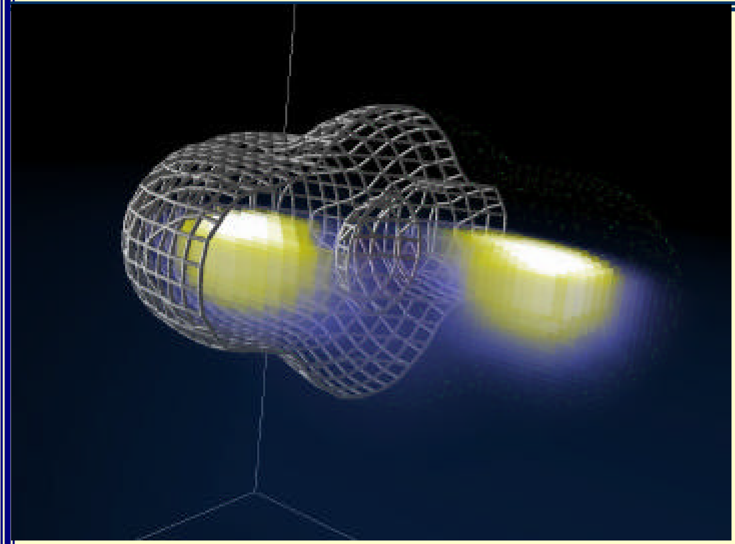
- For SX6 MPI synch, low overhead but increases dramatically w/ 8 procs
- OpenMP Thread Spawn, SX6 lowest overhead & least perf degradation
- OpenMP Scalar Reduction, Power4 fastest up to 8 procs, but with fully loaded SMP SX6 outperforms Pwr3/4 by factors of 2.5x and 6.3x
- Results show Power3/4 does not effective utilize whole SMP

Scientific Kernels: NPB

P	CG				FT				BT			
	Power 3		SX6		Power 3		SX6		Power 3		SX6	
	Mflop/s	%L1	Mflop/s	AVL	Mflop/s	%L1	Mflop/s	AVL	Mflop/s	%L1	Mflop/s	AVL
1	54	68	470	199	133	91	2021	256	144	96	3693	100
2	55	72	260	147	120	91	1346	256				
4	54	73	506	147	117	92	1324	255	127	97	2395	51
8	55	81	131	117	112	92	1241	254				
16	48	86			95	92			102	97		

- CG on SX6, low perf due to bank conflicts & low AVL and low VOR (95%)
- FT on SX6 3 lines of code change (increase AVL), over 10x spdup vs Pwr3
- BT inline routines (necessary for vector) and manual expansion small loops
  Impressive serial perf (26x Pwr3). Larger P reduced AVL due to 3D decomp
  Poor Pwr3 16 proc perf due to large number of synchs

Astrophysics: CACTUS



- Numerical solution of Einstein's equations from theory of general relativity
- Set of coupled nonlinear hyperbolic & elliptic systems with thousands of terms
- CACTUS evolves these equations to simulate high gravitational fluxes, such as collision of two black holes
- Uses ADM formulation: domain decomposed into 3D hypersurfaces for different slices of space along time dimension
- Examine two versions of core CACTUS ADM solver:
  - BenchADM: older F77 based computationally intensive, 600 flops per grid point
  - BenchBSSN: (serial version) newer F90 solver intensive use of conditional statements in inner loop

CACTUS: Porting Details

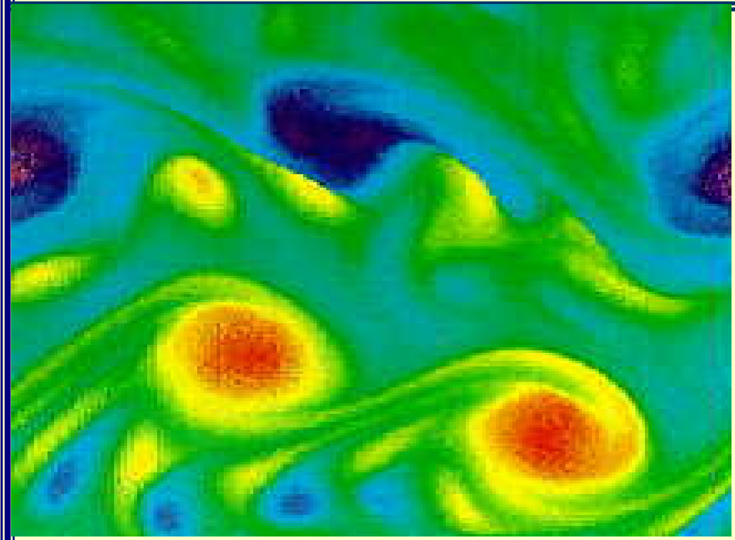
- BenchADM only required compiler flags, but vectorized only on innermost loop (***x,y,z***)
- Increasing x-dimension improved AVL and performance
- BenchBSSN: Poor initial vector performance
- Loops nest too complex for auto vectorization
- Explicit vectorization directives unsuccessful
- Diagnostic compiler messages indicated (false) scalar inter-loop dependency
- Converted scalars to 1D temp arrays of vector length (256)
- Increased memory footprint, but allowed code to vectorize

CACTUS: Performance

Code	P	Size	Power 3		Power4		SX6		
			Mflop/s	%L1	Mflop/s	%L1	Mflop/s	AVL	VOR
ADM	1	127^3	274	99.4	672	92.2	3912	127	99.6%
ADM	8	127^3	251	99.4	600	92.4	2088	127	99.3%
ADM	16	127^3	223	99.4	538	93.0			
ADM	32	127^3			380	97.0			
BSSN	1	80x80x40	209		547		1765	80	99%

- BenchADM: SX6 achieves 14X and 6X speedup over Power3/4
  SX6's 50% of peak is highest achieved for this benchmark
- BenchBSSN: SX6 is 8.4X and 3.2X faster than Power3/4 (80x80x40)
- Lower SX6 performance due to conditional statements
  Power3/4 performance improves w/ small problems (opposite SX6)
- Strong correlation between AVL and SX6 performance (long vectors)

Plasma Fusion: TLBE



- TLBE uses a Lattice Boltzmann method to model turbulence and collision in fluid
- Performs 2D simulation of high temperature plasma using hexagonal lattice and BGK collision operator
- Pictures shows vorticity contours in 2D decay of shear turbulence from TLBE calc

- Three computational components:
  - Integration - Computation of mean macroscopic variable (MV)
  - Collision - Relaxation of MV after colliding
  - Stream - Propagation of MV to neighboring grid points
- First two steps good match for vector - each grid point computed locally Third step requires strided copy
- Distributing grid w/ 2D decomp for MPI code, boundary comm for MV

TLBE: Porting Details

- ✍ Slow initial performance using default (-C opt) & aggressive (-C hopt) compiler flags 280Mflops
- ✍ Flow trace tool (ftrace) showed 96% of runtime in collision
- ✍ AVL of 6: vectorized along inner loop of hexagonal directions, instead of grid dimensions
- ✍ Collision routine rewritten using temporary vectors and switched order of two loops to vectorize over grid dim
- ✍ Inserted new collision into MPI code for parallel version

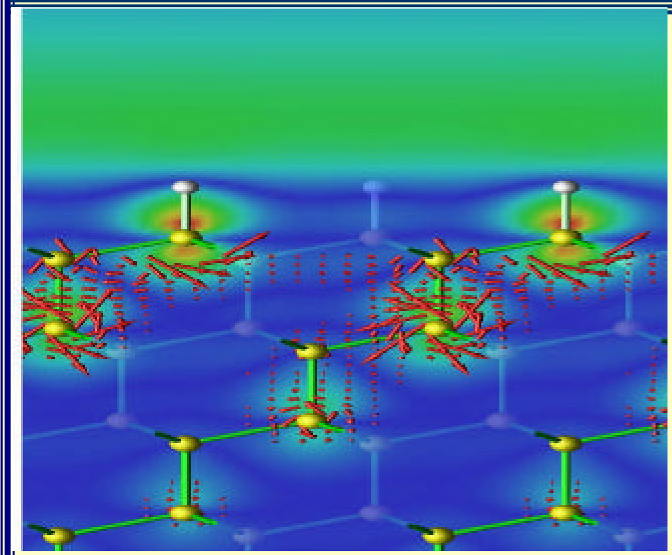
TLBE: Performance

P	Power 3		Power4		SX6		
	Mflop/s	%L1	Mflop/s	%L1	Mflop/s	AVL	VOR
1	70	50	250	58	4060	256	99%
2	110	77	300	69	4060	256	99%
4	110	75	310	72	3920	256	99%
8	110	77	470	87	3050	255	99%
16	110	73	360	89			
32			440	89			

2048x 2048 Grid

- SX6 28x and 6.5x faster than Power3/4 with minimal porting overhead
- SX6 perf degrades w/ 8 procs: bandwidth contention & synch overheads
- Power3/4 parallel perf improves due to improved cache (smaller grids)
- Complex Power4 behavior due to 3-level cache and bandwidth contention

Material Science: PARATEC



- PARATEC performs first-principles quantum mechanical total energy calculation using pseudopotential & plane wave basis set
- Density Functional Theory to calc structure & electronic properties of new materials
- DFT calc are one of the largest consumers of supercomputer cycles in the world
- PARATEC uses all-band CG approach to obtain wavefunction of electrons
- Part of calc in real time other in Fourier space using specialized 3D FFT to transform wavefunction
- Code spends most time in vendor supplied BLAS3 and FFTs
- Generally obtains high percentage of peak on different platforms
- MPI code divides plane wave components of each electron across procs

PARATEC: Porting Details

- Compiler incorrectly vectorized loops w/ dependencies "NOVECTOR" compiler directives were inserted
- Most time spent in BLAS3 and FFT, simple to port
- SX6 BLAS3 efficient with high vectorization
- Standard SX6 3D FFT (*ZFFT*) ran low percentage of peak
- Necessary to convert 3D FFT to simultaneous 1D FFT calls (vectorize across the 1D FFTs)

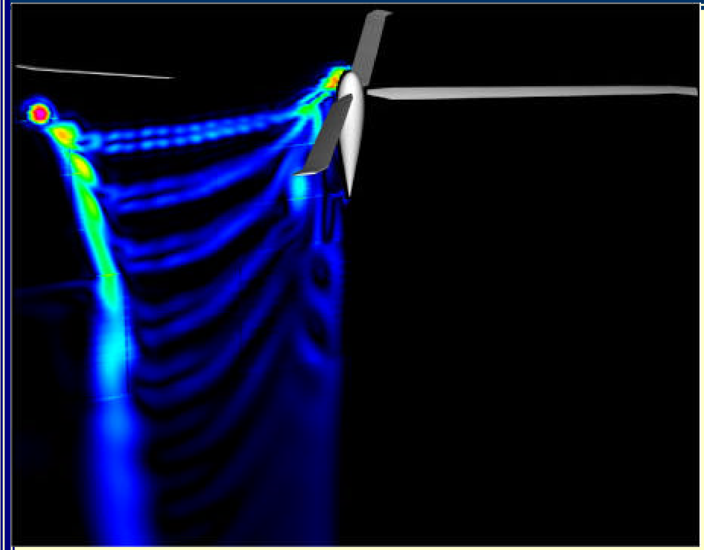
PARATEC: Performance

P	Power 3	Power4	SX6		
	Mflop/s	Mflop/s	Mflop/s	AVL	VOR
1	915	2290	5090	113	98%
2	915	2250	4980	112	98%
4	920	2210	4700	112	98%
8	911	2085	4220	112	98%
16	840	1572			
32		1327			

250 Si-atom system w/ 3 CG steps

- PARATEC vectorizes well (64% peak on 1 P) due to BLAS3 and 3D FFT
- Loss in scalability due to initial code set up (I/O etc) – that does not scale
- Performance increases with larger problem sizes and more CG steps
- Power3 also runs at high efficiency (61% on 1 P)
- Power4 runs at 44%, and perf degrades due to poor flop/bdwth ratio
  However 32 SMP Power4 exceeds performance of 8 SMP SX6

Fluid Dynamics: OVERFLOW-D



- OVERFLOW-D overset grid method for high-fidelity Navier Stokes CFD simulation
- Viscous flow simul for aerospace config
- Can handle complex designs with multiple geometric components
- Flow eqns solved independ on each grid, boundary values in overlap then updated
- Finite difference in space, implicit/explicit time stepping
- Overlapping boundary points updated using a Chimera interpolation
- Code consists of outer "time-loop" and inner "grid-loop"
- MPI version based on multi-block serial code (block groups per proc)
- Hybrid paradigm exploits second level of parallelism
- OpenMP directives used within grid loop (comp intensive section)

OVERFLOW-D: Porting Details

- Original code was designed to exploit vector arch
- Changes for SX6 made only in linear solver: LU-SGS combines LU factorization and Gauss-Siedel relaxation
- Changes dictated by data dependencies of solution process
- On IBM a pipeline strategy was used for cache reuse
- On SX6 a hyper-plane algorithm was used for vectorization
- Several other code mods possible to improve performance

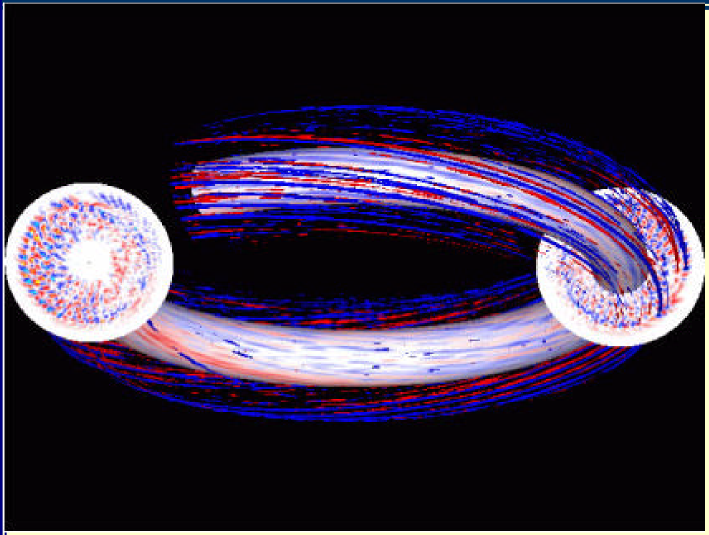
OVERFLOW-D: Performance

8 million grid points 10 time steps

P	Power 3		Power4	SX6		
	sec	%L1	sec	sec	AVL	VOR
2	47	93	16	5.5	87	80%
4	27	95	8.5	2.8	84	76%
8	13	97	4.3	1.6	79	69%
16	8	98	3.7			
32			3.4			

- SX6 8 processor time less than one half 32 processor Power4
- Scalability similar for both architectures due to load imbalance
- SX6 low AVL and VOR explain max of only 7.8 Gflop/s on 8 procs
- Reorganizing code through extensive effort would improve SX6 perf
- SX6 outperforms Power4 for both MPI and hybrid (not shown)
- Hybrid increased complexity with little performance gain – however can help with load balancing (when few blocks relative to procs)

Magnetic Fusion: GTC



- Gyrokinetic Toroidal Code: transport of thermal energy (plasma microturbulence)
- Goal is burning plasma power plant producing cleaner energy
- GTC solves gyroaveraged gyrokenetic system w/ particle-in-cell approach (PIC)
- PIC scales N instead of N^2 – particles interact w/ electromag field on grid
- Allows eqns of particle motion solved with ODEs (instead of nonlinear PDEs)
- Main computational tasks:
  - Scatter: deposit particle charge to nearest grid points
  - Solve the Poisson eqn to get potential at each grid point
  - Gather: Calc force on each particle based on neighbors potential
  - Move particles by solving eqn of motion
  - Find particles moved outside local domain and update
- Expect good parallel performance since Poisson eqn solved locally

GTC: Porting Details

- Initially compilation produced poor performance
  - Nonuniform data access and many conditionals
- Necessary to increase “loop count” compiler flag
- Removed I/O from main loop to allow vectorization
- Compiler directed loop fusion helped increase AVL
- Bottleneck in scatter operation: many particles deposit charge to same grid point causing memory dependence
- Each particle writes to local temp array (256) – no depend
- Arrays merged at end of computation
- No depend, but increase mem traffic and reduced flop/byte

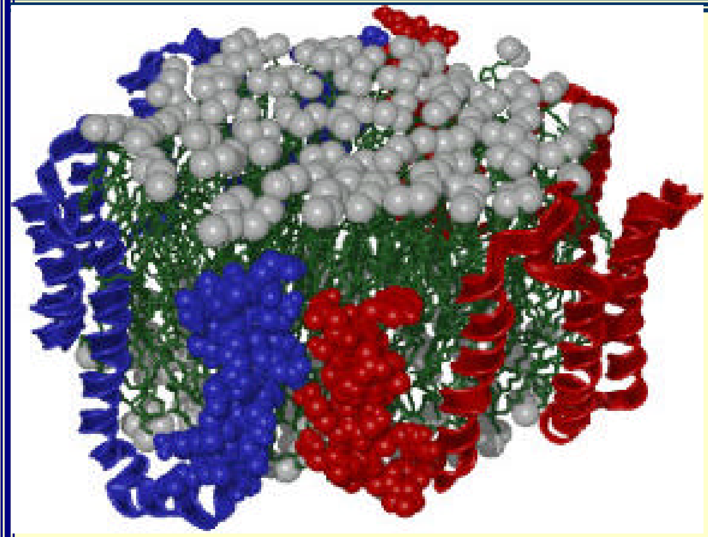
GTC: Performance

4 million particles
301,472 grid pts

P	Power 3		Power4		SX6		
	Mflop/s	%L1	Mflop/s	%L1	Mflop/s	AVL	VOR
1	153	95	277	89	701	187	98%
4	163	96	310	91	548	182	98%
8	167	97	326	92	391	175	97%
16	155	97	240	93			
32			275	93			

- Modest 9% peak SX6 serial performance (2.7x and 5.3x faster Power3/4)
- Scalar units need to compute indices for indirect addressing
- Scatter/gather required for underlying unstructured grid
  - Also at odds with cache based architecture
- Although scatter routine "optimized" running at only 7% peak
- Extensive algorithmic & implem work required for high performance

Molecular Dynamics: Mindy



- Simplified serial molecular dynamics C++, derived from parallel NAMD (Charm++)
- MD simulations infer functions of biomolecules from their structures
- Insight to biol process & aids drug design
- Mindy calc forces between N atoms via Particle Mesh Ewald method O(NlogN)
- Divide into boxes, comp electrostatic interaction w/ neighbor boxes
- Neighbor lists and cutoffs used to decrease # of force calcs
- Reduction of work from N^2 to NlogN causes:
  - Increase branch complexity
  - Nonuniform data access

Mindy: Porting Details

- Uses C++ objects: compiler hindered in ability to vectorize
  - Aggregate date types call member functions
  - Compiler directive (no dep) used, but w/ limited success
- Two optimization strategies, NO_EXCLUSION & BUILD_TMP
- NO_EXCLUSION: Decrease # of conditionals & exclusions
  - Increase vol of comp but reduces inner-loop branching
- BUILT_TMP: Gen temp list of inter-atom forces to comp
  Then compute force calc on list with vectorized loop
  - Increase comp & requires extra mem (reduce flop/byte)

Mindy: Performance

Power 3	Power4	SX6: NO_EXCL			SX6: BUILD_TMP		
sec	sec	sec	AVL	VOR	sec	AVL	VOR
15.7	7.8	19.7	78	0.03%	16.1	134	35%

922224 atom system

- Poor SX6 performance (2% of peak), half speed of Power4
- NO_EXCL: Small VOR, all work performed in scalar unit (1/8 of vec unit)
- BUILD_TMP: Increased VOR, but increased mem traffic for temp arrays
- This class of app at odds w/ vectors due to irregular code structure
- Poor C++ vectorizing compiler –difficult to extract data-parallelism
- Effective SX6 use requires <u>extensive</u> reengineering of algorithm and code

Application Summary

Name	Discipline	Lines Code	P	Pwr3	Pwr4	SX6	SX6 speedup vs	
				% Pk	%Pk	%Pk	Pwr3	Pwr4
TLBE	Plasma Fusion	1500	8	7	9	38	28	6.5
Cac-ADM	Astrophys	1200	8	17	12	26	14	5.8
Cac-BSSN	Astrophys	8200	1	14	11	22	8.4	3.2
OVER-D	Fluid Dynam	100000	8	8	7	12	8.2	2.7
PARATEC	Mat Science	50000	8	61	40	53	4.6	2.0
GTC	Magn Fusion	5000	8	11	6	5	2.3	1.2
MINDY	Molec Dynam	11900	1	6	5	2	1.0	0.5

- Comp intensive CAC-ADM only compiler directives (14x P3 speedup)
- CAC-BSSN, TLBE, minor code changes for high % peak
- OVER-D substantially diff algorithmic approach, fair perf on both arch
- PARATEC relies on BLAS3 libraries, good performance across all arch
- GTC and Mindy poor vector performance due to irregular comp

Summary

- Microbenchmarks: specialized SX6 vector/memory significantly outperform commodity-based superscalar Power3/4
- Vector optimization strategies to improve AVR and VOR
  - Loop fusion/reordering (explicit /compiler directed)
  - Intro temp variables to break depend (both real & compiler imagined)
  - Reduction of conditional branches
  - Alternative algorithmic approaches
- Vectors odds with many modern sparse/dynamic codes
  - Indirect addr, cond branches, loop depend, dynamic load balancing
- Direct all-to-all methods may be ineffective at petascale
- Modern C++ methods difficult to extract data parallel
- Vectors specialized arch extrem effective for restricted class of apps

Future work

- Develop XTREAM benchmark to examine microarchitecture characteristics and compiler performance
- Develop SQMAT microbenchmark, tunable computational intensity and irregularity
- Examine key scientific kernels in detail
- More applications: Climate, AMR, Cosmology
- Leading architectures: ES, X1, EV7
- Future arch of various comp granularities w/ new interconn
  - Red Storm, Bluegene/*

Extra Slides

CACTUS: Performance

Serial Code	Problem Size	Power 3	Power4	SX6		
		Mflop/s	Mflop/s	Mflop/s	AVL	VOR
Bench ADM	128x128x128	34	316	4400	127	99%
Bench BSSN	128x128x64	186	1168	2350	128	99%
	80x80x40	209	547	1765	80	99%
	40x40x20	249	722	852	40	98%

- BenchADM: SX6 achieves 129X and 14X speedup over Power3/4! SX6's 55% of peak is highest achieved for this benchmark
- BenchBSSN: SX6 is 8.4X and 3.2X faster than Power3/4 (80x80x40)
- Lower SX6 performance due to conditional statements
- Strong correlation between AVL and SX6 performance (long vectors)
- Power3/4 performance improves w/ smaller problem size (unlike SX6)